

## ANALYTIC MODELS OF MULTITASK PROCESSES\*

Timothy L. Johnson

Lecturer  
Department of Electrical  
Engineering & Computer Science  
Massachusetts Institute of Technology  
Cambridge, Massachusetts 02139

Senior Scientist  
Bolt, Beranek and Newman, Inc.  
50 Moulton Street  
Cambridge, Massachusetts 02238

Abstract

Asynchronous multitask processes occur in a wide variety of control applications ranging from industrial control to computer operating systems, yet no analytical methods are available for studying their detailed behavior. The preliminary results reported here illustrate that a very general class of such processes can be represented by discontinuous hybrid-state discrete-time systems.

Background and Motivation

A multitask process is characterized by a number of tasks which operate concurrently or sequentially, on an external resource or data base. The timing of the tasks is generally asynchronous in that new task execution is initiated by the completion of previous tasks. If necessary, synchrony and sequential ordering of tasks can be enforced in a number of ways through the task definitions themselves. However in this research no such constraints are imposed: rather, the general qualitative behaviors which may arise in such systems are analyzed. Only two basic assumptions are imposed: (1) a task requires a finite amount of time and storage to execute, and (2) task descriptions are fixed, in that the execution of a task cannot alter its own nature nor the number or nature of any other tasks.

The range of possible behavior of such systems is so large that the problem of conceptualizing, analyzing and "debugging" multitask processes is very common and enormously complex. In addition to deterministic processor scheduling [9], two approaches are presently in use: stochastic queueing analysis [1], [2] and simulation [3], [4]. Queueing analysis is most useful for evaluating the average performance properties of an operational multitasking system, while simulation allows certain undesirable properties of a planned system to be discovered and corrected during the design process. Neither of these methods provides very much

\*This research has been performed at the M.I.T. Laboratory for Information and Decision Systems with support provided by the U.S. Air Force Office of Scientific Research under Contract F49620-80-C-0002. The results presented here do not necessarily represent the views of the U.S. Government.

insight about generic problems in the design of such systems, nor do they provide ideas about how to remedy or detect flaws. The results reported here constitute a modest step in that direction. They represent a generalization of many models of deterministic processor scheduling in that the tasks may have feedback interactions and that scheduling and timing may be interdependent.

Model Development

Let  $t \in [t_0, \infty)$  denote time. Three sets of state variables will be identified:

- $x^1$  - those states which vary continuously with time and take on real values.
- $x^2$  - those states which are real-valued but change only at discrete instants of time.
- $x^3$  - those states which are discrete-valued and (necessarily) change only at discrete instants of time.

The state set is denoted  $X = \{x^1, x^2, x^3\}$ . It will be assumed that these subsets of states are finite-dimensional and readily distinguished.

Let the increasing sequence  $\{t_k\}$  denote the set of all values of  $t$  for which changes in at least one element of  $x^2$  or  $x^3$  occur, and let the values of the states prior to and following  $t_k$  be denoted  $x_k^{i-}$ ,  $x_k^{i+}$ , respectively, for  $i = 1, 2, 3$ . In the sequel,  $x_k^{i+}$  will be identified with  $x_k^i$ . The instants  $\{t_k\}$  will be identified with task initiation or termination times. Let the (finite) set of tasks in the system be denoted  $G = \{G_1 \dots G_n\}$ . Associated with each task is an initiation function, a termination function and a state-update function<sup>1</sup>:

- $g_j^I: X \rightarrow \{0,1\}$  - initiation function for task  $j$
- $g_j^T: X \rightarrow \{0,1\}$  - termination function for task  $j$
- $f_j: X \rightarrow X$  - state-update function for task  $j$

<sup>1</sup>To simplify this exposition, these are assumed to be time-invariant; however, this assumption may be relaxed.

Each task is either "on" or "off": let  $\check{G}$  denote those which are off, so that  $G = \hat{G} \cup \check{G}$  and  $\hat{G} \cap \check{G} = \emptyset$  (the null set). The subscript  $\hat{j}$  will be used to denote tasks which are "on" and  $\check{j}$  to denote tasks which are "off". The task succession rule is as follows: A transition time,  $t_k$ , is declared whenever

(a) For some  $j \in \check{j}$ ,  $g_j^I(x^1, x^2, x^3)$  undergoes a 0→1 transition

or

(b) For some  $j \in \hat{j}$ ,  $g_j^T(x^1, x^2, x^3)$  undergoes a 0→1 transition

Between task transition times, only the states  $x^1$  can change, according to a state equation

$$\dot{x}^1(t) = f(x^1(t), x_k^{2+}, x_k^{3+}) \quad (1)$$

with  $x^1(t_k) = x_k^{1+}$ . At the completion time  $t_k^-$  of a task  $j$ , the transformation

$$\begin{bmatrix} x_k^{1+} \\ x_k^{2+} \\ x_k^{3+} \end{bmatrix} = f_j \left( \begin{bmatrix} x_k^{1-} \\ x_k^{2-} \\ x_k^{3-} \end{bmatrix} \right) \quad (2)$$

is applied, with  $x_k^{2-} \doteq x_{k-1}^{2+}$  and  $x_k^{3-} \doteq x_{k-1}^{3+}$ .

At a transition time, it is possible that more than one task terminates and/or more than one task is initiated. This produces an inherent conflict situation which must be resolved in a consistent manner. For instance, if tasks  $j_1$  and  $j_2$  terminate together, it is not necessarily true that  $f_{j_1} \circ f_{j_2} = f_{j_2} \circ f_{j_1}$  (functional composition may not be commutative). Or if task  $j_1$  is initiated when  $j_2$  terminates, then up-dating with  $f_{j_2}$  may turn off  $j_1$ , while terminating  $j_1$  may turn on  $j_2$  again, etc.

One set of further assumptions which resolves such conflicts is as follows:

- (c) there is a fixed priority among task completions (e.g.  $1 > 2 > 3 > j > \dots > n$ )
- (d) all completions are performed first according to priority, and then initiation functions are re-evaluated to redetermine which tasks (if any) should be initiated at the transition times.
- (e) the minimum completion time of every task is bounded below by a positive number.<sup>2</sup>

<sup>2</sup>This may be achieved by adding to  $x^1$   $n$  "timers". A timer is reset by (2) to the minimum task completion time each time a task is initiated and tested for a positive value in the corresponding task termination function.

Other conflict-resolution methods, such as imposed sequential orderings, are also possible.

Let  $\hat{j}_k \in 2^n$  be the set of tasks active at  $t_k^+$ . Let the transition mapping of (1) be given by  $\phi$ :

$[0, \infty) \times X^1 \rightarrow X^1$ , so that the solution of

$$\dot{x}^1(t) = f(x^1, x_k^{2+}, x_k^{3+}); \quad x^1(t_k) = x_k^{1+} \quad (3)$$

is

$$x^1(t) = \phi(t - t_k, x_k^{1+}; x_k^{2+}, x_k^{3+}) \quad (4)$$

where  $x_k^{2+}, x_k^{3+}$  are viewed as parameters. Define the positive function  $\tau: 2^n \times X \rightarrow R^+$  to be the first transition-time encountered with processes  $\hat{j} \in 2^n$  active at  $t = t_k$ , with initial state  $x = (x^1, x^2, x^3) \in X$ . This can be tabulated by integrating (1) and applying rules (a) to (e). Let the function  $\sigma: 2^n \times X \rightarrow 2^n$  define the next set of active tasks, determined from the preceding priority rules, at the transition time defined by  $\tau$ . In other words,

$$t_{k+1} = t_k + \tau(\hat{j}_k, x_k^{1+}, x_k^{2+}, x_k^{3+}) \quad (5)$$

$$\hat{j}_{k+1} = \sigma(\hat{j}_k, x_k^{1+}, x_k^{2+}, x_k^{3+}) \quad (6)$$

The important point to observe is that, in principle, it is not necessary to include the continuous-time part of the dynamics, since  $\tau$  and  $\sigma$  can be pre-computed from  $f$ ,  $\{g_j^I\}$ , and  $\{g_j^T\}$ .

In summary, the dynamics of the asynchronous multitask system can always be represented in the form

$$\left. \begin{aligned} t_{k+1} &= t_k + \tau(\hat{j}_k, x_k) \\ x_{k+1} &= f_{\hat{j}_k}(x_k) \\ \hat{j}_{k+1} &= \sigma(\hat{j}_k, x_k) \end{aligned} \right\} \quad (7)$$

where  $f_{\hat{j}_k}$  is the composition, according to priority, of the transition functions (2) of the tasks completing at  $t_{k+1}$ . It is then clear that  $t_k$  may be combined with  $x_k^{2+}$ , and that  $x_k^{3+}$  may be combined with  $\hat{j}_k$  to yield a general discontinuous hybrid discrete-time system. Extensions to stochastic behavior of  $f$ ,  $\{f_j\}$ ,  $\{g_j^I\}$  and  $\{g_j^T\}$  are readily accommodated.

#### Qualitative Properties

The finite-state part of (7) may be further aggregated to produce an equivalent real-state discrete-time system with discontinuous transition function. In other special cases a purely finite-state system may be derived. Systems of this gen-

eral class have been discussed by Johnson [5] and Kaliski and Lemone [6]. Their behavior may roughly approximate the behavior of discontinuous systems discussed in Utkin [7] and Johnson [8]. A possible behavior in this situation is a discrete-time approximation to sliding mode behavior, which is closely akin to the phenomenon of "thrashing" that has been observed in heavily-loaded multitasking systems. Types of stability have been defined in [8].

### Conclusions

It has been shown that a broad class of real-time multitasking systems may be represented as discrete-time (discontinuous) finite dimensional dynamic systems. Some issues which remain to be investigated are: (1) consistency requirements for initiation and termination functions, (2) alternative forms of conflict resolution (e.g., other than priority assignment), (3) minimal realizations and canonical forms, (4) development of design synthesis tools to obtain guaranteed performance.

### References

- [1] M.G. Kienzle and K.C. Sevcik, "Survey of Analytic Queueing Network Models of Computer Systems", Proc. 1979 Conf. on Simulation, Measurement and Modelling of Computer Systems, pp. 113-129.
- [2] G.K. Hutchinson and J.J. Hughes, "A Generalized Model of Flexible Manufacturing Systems", Proc. Multi-Station Digitally-Controlled Mfg. Systems Workshop, Milwaukee, Wisc., Jan. 1977; pp. 88-115.
- [3] Shannon, R.E., Systems Simulation: The Art and Science, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1975.
- [4] Pritsker, A.A.B., The GASP IV Simulation Language, J. Wiley & Sons, New York, 1974.
- [5] Johnson, T.L., "Finite-State Compensators for Continuous Processes", Proc. 7th IFAC Congress, Helsinki, Finland, June 1978, pp. 1823-1828.
- [6] Kaliski, M.E. and Lemone, K., "Discrete Codings of Continuous Valued Signals", Proc. 14th Ann. Conf. on Info. Sci. and Sys., Johns Hopkins Univ., March 1980.
- [7] Utkin, V.I., Sliding Modes and Their Application to Valuable-Structure Systems, MIR Publishers, Moscow, 1978.
- [8] Johnson, T.L., "Stability of Diced Systems", Proc. 19th IEEE Conf. on Decision and Control, Albuquerque, N.M., Dec. 1980.
- [9] Gonzalez, M.J., "Deterministic Processor Scheduling", ACM Computing Surveys, Vol. 9, No. 3, pp. 173-204.